Final Report

A Skill-centric Hierarchical Framework for Scalable Robot Tasks

Contributions:

Dong Fang (223040011) : Experiments

Yuming Liu (223015138) : Coding

Jiawei Gu (223040015) : Researchs

Repo : https://github.com/handleandwheel/manipulation_final_proj

School of Science and Engineering

The Chinese University of Hong Kong, Shenzhen

2025 Term 2

Final Report

1. Motivation:

Recent advancements in vision-language models (VLMs) have enabled the development of new vision-language-action (VLA) frameworks. These frameworks combine the ability to understand visual information with the ability to predict actions guided by language. End-to-end approaches based on VLA frameworks have shown promising results in manipulation tasks. However, as illustrated in Fig. 1(a), their task-centric design creates significant challenges when applied to open-world scenarios.

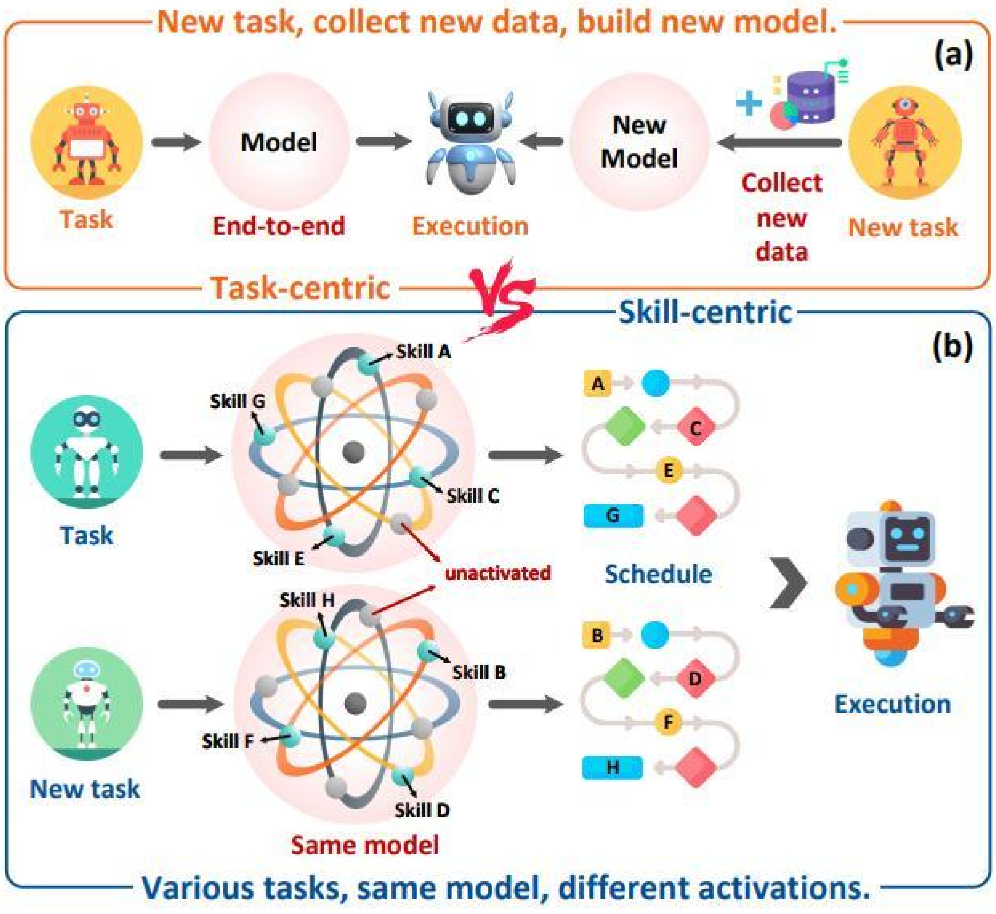


Figure 1. Task-Centric vs. Skill-Centric.

One major issue is that these frameworks require complete demonstrations of tasks. This means that as tasks become more complex, the amount of data needed grows rapidly, leading to exponential increases in data requirements. Another issue is that these end-to-end architectures struggle to handle new combinations of tasks. They are not flexible enough to adapt to novel or unseen task compositions. Additionally, these systems rely on black-box learning mechanisms, which make it difficult to identify the source of errors or diagnose problems when they arise. These limitations are caused by the way these frameworks combine three core robotic abilities into a single process.

2. Challenges

The main challenge for open-world manipulation lies in the ability to generalize to new scenarios and tasks. This requires the flexible recombination of meta-skills to adapt to new task specifications. However, most existing methods fail to achieve this level of generalization. Current solutions can be grouped into two main categories.

- The first category includes task-specific traditional methods, such as imitation learning (IL) and reinforcement learning (RL). These methods closely link the perception and action spaces, creating a rigid connection between what the system sees and how it acts. Because of this tight coupling, these methods completely fail when faced with new object-task pairings.
- The second category includes methods that rely on large language models (LLMs) . These methods leverage the vast amount of prior knowledge embedded in LLMs to guide decision-making. While they represent a more flexible approach, they still suffer from significant limitations.
- Another major issue is the difficulty in identifying and fixing errors within these systems. The complexity of LLM-based methods makes debugging and system improvement challenging, which further limits their ability to handle diverse and unpredictable environments. These problems make it hard for both categories of methods to function effectively in open-world scenarios.

3. Methodology

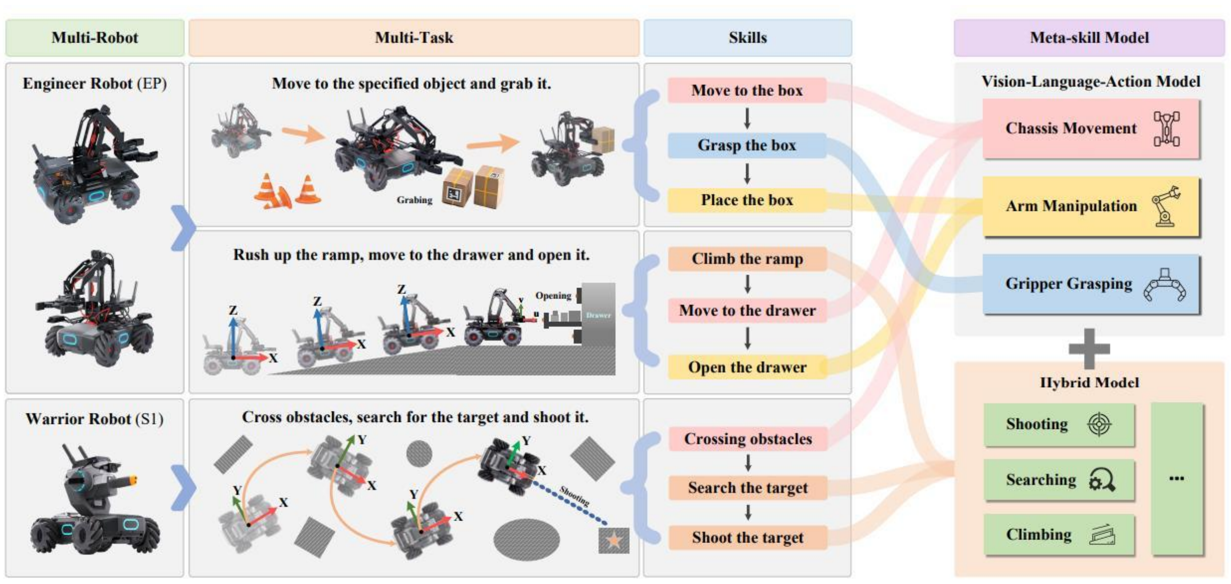


Figure 2. Inspiration of the skill-centric method.

Our main idea is that separating skill learning from task composition makes it possible to:

(a) reuse skills for different tasks;

(b) identify errors more easily;

(c) adapt to new tasks with less data.

To address these issues, we introduce ours Robomatrix. This is a skill-focused framework with a hierarchical structure. It allows tasks to be completed by combining different skills.

As shown in Fig. 1(b), our fully trained VLA skill model can handle new tasks without additional training or collecting more data. It does this by activating specific skill responses, like "Grasp-Response" or "Move-Response," based on what it sees in the environment and the task it needs to perform. This means ours can complete tasks by rearranging and recombining skills, so there is no need to train the model again for each new task.

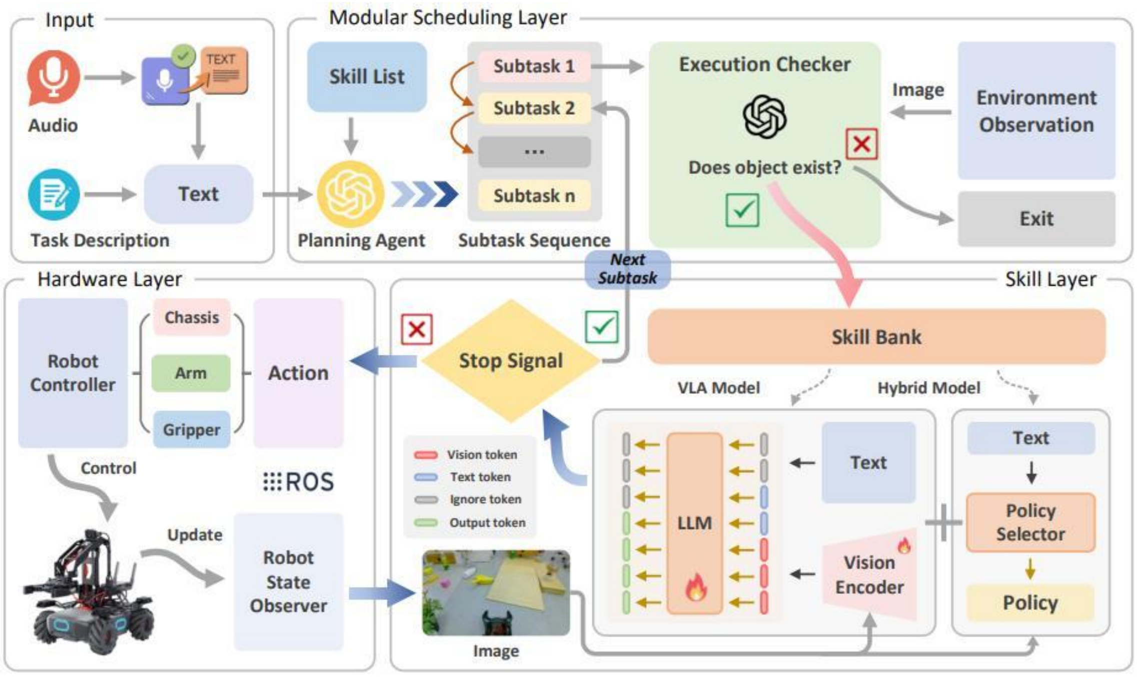


Figure 3. ours Overview.

To make this work, ours mothod uses a hierarchical system for breaking down tasks and organizing the necessary skills. The framework is divided into three parts: thescheduling layer, the skill layer, and the hardware layer. Each part has a specific role in ensuring that tasks are completed efficiently.

- The scheduling layer is the first part of the framework. It uses a general-purpose large language model (LLM) to break the task into smaller steps. It also decides which skill models are required to complete the task.
- The skill layer holds the meta-skill models. These models are pre-trained and can perform specific actions. Each meta-skill is designed to handle a certain type of problem or task.
- The third part of the framework is the hardware layer. This layer includes the physical robot and the communication system. The robot is equipped with sensors and tools that allow it to interact with its environment. The communication system ensures that the robot can receive instructions from the higher layers and send feedback to them.

The interaction between these three layers ensures that tasks are completed smoothly.

For instance, when a task is assigned, the scheduling layer first breaks it into smaller steps. Then, it selects the necessary skills from the skill layer. Next, the hardwarelayer executes the actions required to complete the task. If something goes wrong, the robot communicates the issue back to the scheduling layer, which adjusts the plan and tries again.

4. Experiments

4.1 Implementation Details

4.1.1 Dataset and Annotation

We use DJI's RoboMaster series robots as the physical platform for ours. These robots provide a reliable and flexible base for the system. Different types of robots can be connected to a single computer. They communicate with the computer using a specific network communication protocol. This allows ours to control multiple robots at the same time. It makes it easier to manage and coordinate tasks across different robots.

To improve the system, we reorganize the open-source API of RoboMaster. This is done within the framework of Robot Operating System 2 (ROS2). By doing this, the system becomes more flexible and efficient. It allows distributed control, meaning that different parts of the system can work together smoothly. It also improves the scheduling of skill models, which helps the robots complete tasks more effectively. The control mode of the system is designed to be very simple to change. By adjusting the mapping of the control signal source, the control method can be switched easily. For example, the robot can be controlled manually using an Xboxcontroller. This is useful when human intervention is needed for specific tasks. Alternatively, the robot can operate autonomously through a skill model. In this mode,

the robot performs tasks on its own, using the skills it has been trained to execute. This flexibility makes the system suitable for a wide range of applications, from manual operation to fully autonomous tasks.

4.1.2 Dataset and Annotation

We collect data for eight skills from about 5,000 episodes of human demonstrations. These demonstrations show how to perform long-horizon tasks with high quality. To process the data, we use a mix of rule-based and manual annotation. The proportions of these methods are carefully chosen to ensure the accuracy of the annotations. Fig. 4 shows the eight meta-skills used in our VLA model. Each skill can be performed on its own or combined with others to complete long-horizon tasks.

4.2 Performance on Meta-skills

We evaluate eight meta-skills with the VLA model. The evaluation is conducted 10 times for each experiment unless stated otherwise. The bar chart in Fig. 6 shows the results. For seen objects and seen scenes, the results demonstrate that the VLA model performs very well. The results for unseen objects and unseen scenes also show strong performance. This proves that the model has good generalization ability.

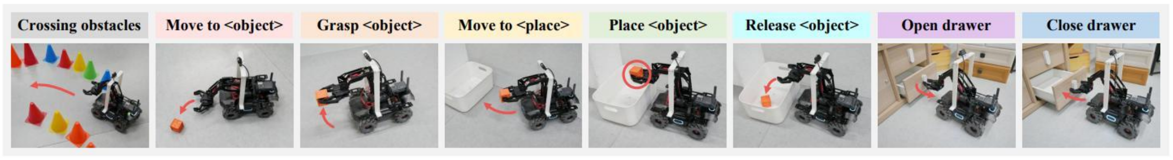


Figure 4. Illustration of meta-skills in the VLA model.

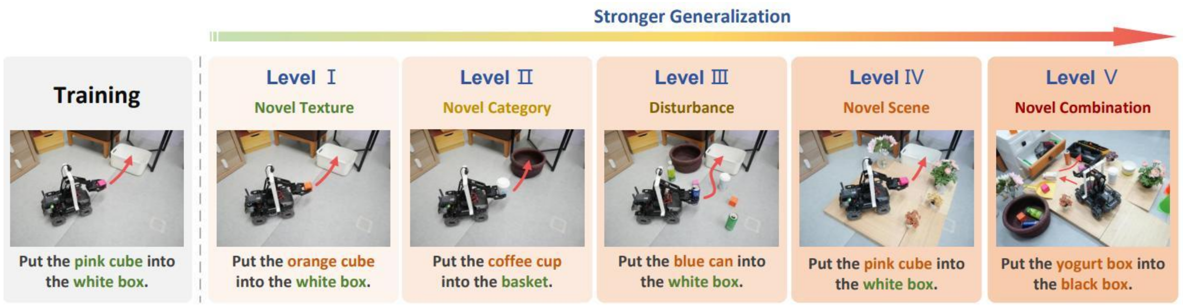

Stronger Generalization
Training
Put the pink cube into the white box.
Level Ⅰ
Novel Texture
Put the orange cube into the white box.
Level Ⅱ
Novel Category
Put the coffee cup into the basket.
Level Ⅲ
Disturbance
Put the blue can into the white box.
Level Ⅳ
Novel Scene
Put the pink cube into the white box.
Level Ⅴ
Novel Combination
Put the yogurt box into the black box.

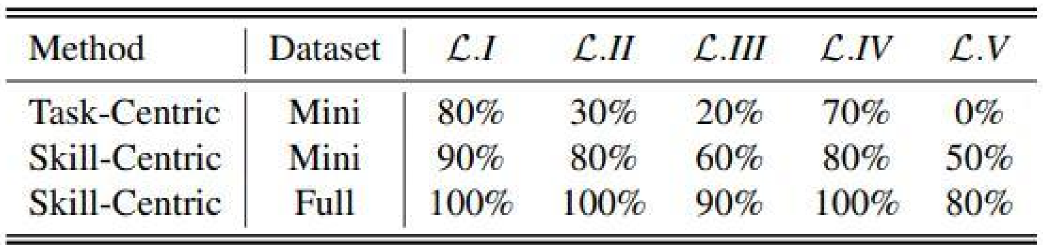
Figure 5. Evaluation Protocol for ours generalization at the object and scene levels.

4.3 Performance on Task-level Generalization

*4.3.1 Evaluation Protocol*Building on VIMA , we propose a 5-level generalization evaluation protocol.

Fig. 5 shows the structure of the protocol. The evaluation focuses on object and scene generalization because open-world environments are complex. Levels I and II measure the difficulty of object generalization. Level III acts as a transition point between object and scene generalization. Levels IV and V focus on scene generalization and test how well the model performs in more complex environments.

4.3.2 Generalization

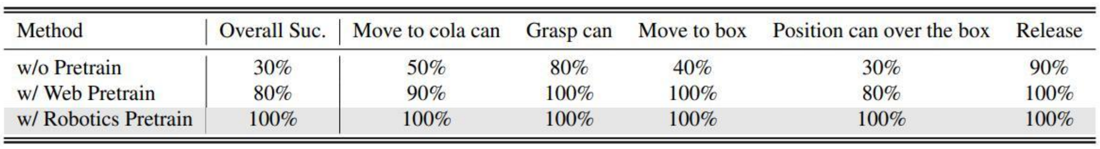


Method	Dataset	$\mathcal{L}.I$	$\mathcal{L}.II$	$\mathcal{L}.III$	$\mathcal{L}.IV$	$\mathcal{L}.V$
Task-Centric	Mini	80%	30%	20%	70%	0%
Skill-Centric	Mini	90%	80%	60%	80%	50%
Skill-Centric	Full	100%	100%	90%	100%	80%

Table 1. Comparison of Task-Centric and Skill-Centric methods.Related Work

For object and scene generalization, Table 1 shows the performance comparison between the task-centric model and our skill-centric VLA model. The comparison is done using both the mini dataset and the full dataset. In simpler levels, our method performs slightly better than the task-centric approach. In more challenging levels, the skill-centric approach performs much better than its counterpart. These results show that the skill-centric model works better for difficult tasks and long-horizon tasks.
For task and embodiment generalization, we run experiments on two types of long-horizon tasks.

4.4 Ablation Study



Method	Overall Suc.	Move to cola can	Grasp can	Move to box	Position can over the box	Release
w/o Pretrain	30%	50%	80%	40%	30%	90%
w/ Web Pretrain	80%	90%	100%	100%	80%	100%
w/ Robotics Pretrain	100%	100%	100%	100%	100%	100%

Table 2. Success rates for each step in a sequential long-horizon task under different pretraining methods.

4.4.1 Pretraining

In Tab. 2, we show three experimental settings to explain why alignment training is necessary and important. These settings are designed to analyze how different training methods affect the performance of the VLA model. The first setting, called “w/o Pretrain,” uses the VLA model with only supervised fine-tuning (SFT) on robot data.This means no alignment training is applied. The model in this setting relies only on robot data for learning.

The second setting, called “w/ web pretrain,” uses a dataset named LLaVA-665K for multi-modal alignment. This alignment training allows the model to learn from both images and text. It helps the model develop a better understanding of how to relate visual and textual data. This improves its ability to generalize across tasks.

The third setting, called “w/ Robotics Pretrain,” combines co-fine-tuning with both LLaVA-665K data and robot skill data. After this, the model undergoes supervised fine-tuning. This setting uses both general multi-modal data and robot-specific data, allowing the model to align within the robot domain while also benefiting from broader alignment knowledge.

The results in Tab. 2 show that multi-modal alignment is very effective. The performance is significantly better when alignment training is included. The third setting, which integrates alignment within the robot domain, achieves the best results. This demonstrates that fine-tuning with robot-specific data further improves the model’s ability to handle tasks. These findings highlight the importance of alignment training in achieving high performance in both general and robotic tasks.

4.4.2 Long-Horizon

Method	Average	Easy	Medium	Hard
Task-Centric	73%	100%	80%	40%
Skill-Centric	**93%**	100%	**100%**	**80%**

Table 3. Ablation Study on Long-Horizon Tasks with Varying Difficulty.

Tab. 3 shows an ablation study that focuses on long-horizon tasks with different difficulty levels. In these tasks, the level of difficulty increases as the task horizon becomes longer. For tasks with a short horizon, which are easier to solve, the success rates of the task-centric and skill-centric methods are very similar. Both methods can handle these simpler tasks effectively, so the performance gap between them is small. For tasks with a medium task horizon, the skill-centric method performs much better than the task-centric method. The success rate of the skill-centric approach is 20%higher in these tasks. This shows that as the tasks become more complex, the skill-centric model begins to show clear advantages.

For hard tasks with a long task horizon, the difference between the two methods becomes even bigger. The skill-centric approach outperforms the task-centric method by 40% in these difficult tasks. This shows that the skill-centric method is much better at handling tasks that require many steps or involve complex decision-making.

As the task horizon increases, the difficulty of the tasks rises, and the advantage of the skill-centric approach becomes even more obvious. The task-centric method struggles to maintain high-performance levels in long-horizon tasks, while the skill-centric method continues to perform well. This makes the skill-centric approach more suitable for challenging tasks that require the execution of multiple steps over a long sequence.

5. Conclusion

In this work, we introduce a skill-centric hierarchical framework for robot task planning and execution. This framework is designed to work in open-world environments. It addresses the need for robots to be both adaptable and efficient when performing tasks in complex and dynamic scenarios. Robots in open-world environments often face challenges where they must handle various objects, execute multiple tasks, and adapt to changing conditions. Our framework is built to handle these challenges effectively by focusing on skill-based task execution.

Overall, the skill-centric hierarchical framework represents a significant advancement in robot autonomy. It combines scalability with strong generalization, making it a powerful tool for addressing the challenges of robot task planning and execution in open-world environments.